LISTING OF CLAIMS

Without prejudice, this listing of the claims replaces all prior versions and listings of the claims in the present application:

1. (Currently Amended) A method for managing bufferpages and redundant copies of records in a local memory associated with a mobile device application, comprising:

(a) retrieving a first record from a remote database memory in response to a request from a first recordset;

(b) saving the first record on a first bufferpage of the local memory associated with the mobile device application, the first bufferpage being associated with the first recordset;

(c) repeating steps (a) and (b) for at least one further record;

(d) when a next record requested by the first recordset is larger than a freespace on the first bufferpage, saving the next record on a second bufferpage of the local memory associated with the mobile device application, the second bufferpage being associated with the first recordset;

(e) determining if one of the first record, the at least one further record, and the next record was previously retrieved and saved in the local memory associated with the mobile device application by at least one of the first recordset and at least one second recordset as one of a prior saved version of the first record, a prior saved version of the at least one further record, and a prior saved version of the next record, respectively; and

(f) storing a pointer with one of the prior saved version of the first record, the prior saved version of the at least one further record, and the prior saved version of the next record, the pointer pointing to the one of the first record, the at least one further record, and the next record if one of the first record, the at least one further record, and the next record was previously retrieved and saved as one of the prior saved version of the first record, the prior saved version

of the at least one further record, and the prior saved version of the next record, respectively, otherwise creating a [[b.o.]] business object kernel pointing to one of the first record, the at least one further record, and the next record.

2. (Canceled)

3. (Original) The method of claim 1, further comprising comparing the freespace on the first bufferpage to a size of the next record.

4. (Withdrawn) A method of managing a memory, comprising:

(a) dividing the memory into a plurality of blocks;

(b) recording in a first block of the memory in a first databuffer at least a first property of a first record in response to a first request of a first recordset;

(c) recording in one of the first block and a second block of the memory in a second databuffer at least one of the first property and a second property of the first record in response to a second request of one of the first recordset and a second recordset; and

(d) storing with the first databuffer a pointer to the second databuffer.

5. (Withdrawn) The method of claim 4, further comprising: recording in one of the first block, the second block, and a third block of the memory in a third databuffer at least one of the first property, the second property, and a third property of the first record in response to a third request of one of the first recordset, the second recordset, and a third recordset; storing with the second databuffer a further pointer to the third databuffer.

6. (Withdrawn) The method of claim 5, further comprising: receiving a signal that the one of the first recordset and the second recordset that provided the second request is inactive; storing with the first databuffer the further pointer to the third databuffer.

7. (Withdrawn) The method of claim 5, further comprising, if the first record stored in response to the first request is a first instance of the first record being stored, creating a kernel pointer in a database access system, the kernel pointer pointing to the first databuffer.

8. (Withdrawn) The method of claim 4, wherein each of the plurality of blocks of memory has a fixed size equal to each other of the plurality of blocks.

9. (Currently Amended) A system for managing bufferpages and redundant copies of records of a mobile device application, comprising:

a remote database memory;

a local program memory associated with the mobile device application; and

a local mobile processor coupled to the remote database memory and the local program memory associated with the mobile device application, the local mobile processor adapted to:

(a) retrieve a first record from the remote database memory in response to a request from a first recordset;

(b) save the first record on a first bufferpage of the local program memory associated with the mobile device application, the first bufferpage being associated with the first recordset;

(c) repeat (a) and (b) for at least one further record;

(d) when a next record requested by the first recordset is larger than a freespace on the first bufferpage, save the next record on a second bufferpage of the local program memory associated with the mobile device application, the second bufferpage being associated with the first recordset;

(e) determine if one of the first record, the at least one further record, and the next record was previously retrieved and saved on the local program memory associated with the mobile device application by at least one of the first recordset and at least one second recordset as one of a prior saved version of the first record, a prior saved version of the at least one

further record, and a prior saved version of the next record, respectively; and

(f) store a pointer with one of the prior saved version of the first record, the prior saved version of the at least one further record, and the prior saved version of the next record, the pointer pointing to the one of the first record, the at least one further record, and the next record if one of the first record, the at least one further record, and the next record was previously retrieved and saved as the prior saved version of the first record, the prior saved version of the at least one further record, and the prior saved version of the next record, respectively, otherwise creating a [[b.o.]] business object kernel pointing to one of the first record, the at least one further record, and the next record.

10. (Currently Amended) A method of managing fixed units of buffer memory associated with a mobile client application, comprising:

retrieving a record stored in a remote database memory;

determining a size of the retrieved record and a size of a freespace of a current fixed unit of buffer memory and:

saving the retrieved record in the current fixed unit of buffer memory if the size of the retrieved record is smaller than the freespace of the current fixed unit of buffer memory;

saving the retrieved record in a next fixed unit of buffer memory if the size of the retrieved record is larger than the freespace of the current fixed unit of buffer memory;

determining if the retrieved record was previously retrieved and stored by the mobile client application and:

storing a pointer pointing from a fixed unit of buffer memory storing a most recent copy of the retrieved record to a fixed unit of buffer memory storing a

new copy of the retrieved record, if the retrieved record was previously retrieved and stored by the mobile client application;

creating a [[b.o.]] business object kernel including a key pointing to the fixed unit of buffer memory storing the new copy of the retrieved record, if the retrieved record was not previously retrieved and stored by the mobile client application.

11. (Previously Presented) The method of claim 1, wherein determining further comprises checking a look-up table.

12. (Previously Presented) The system of claim 9, wherein the local mobile processor is adapted to determine if one of the first record, the at least one further record, and the next record was previously retrieved and saved as the prior record by checking a look-up table.

13. (Previously Presented) The method of claim 10, wherein determining if the retrieved record was previously retrieved and stored by the mobile client application comprises checking a look-up table.

14. (Currently Amended) The method of claim 10, further comprising storing the [[b.o.]] business object kernel in a look-up table.

15. (Previously Presented) The method of claim 10, wherein the key comprises a counter indicating a number of times the retrieved record is stored.